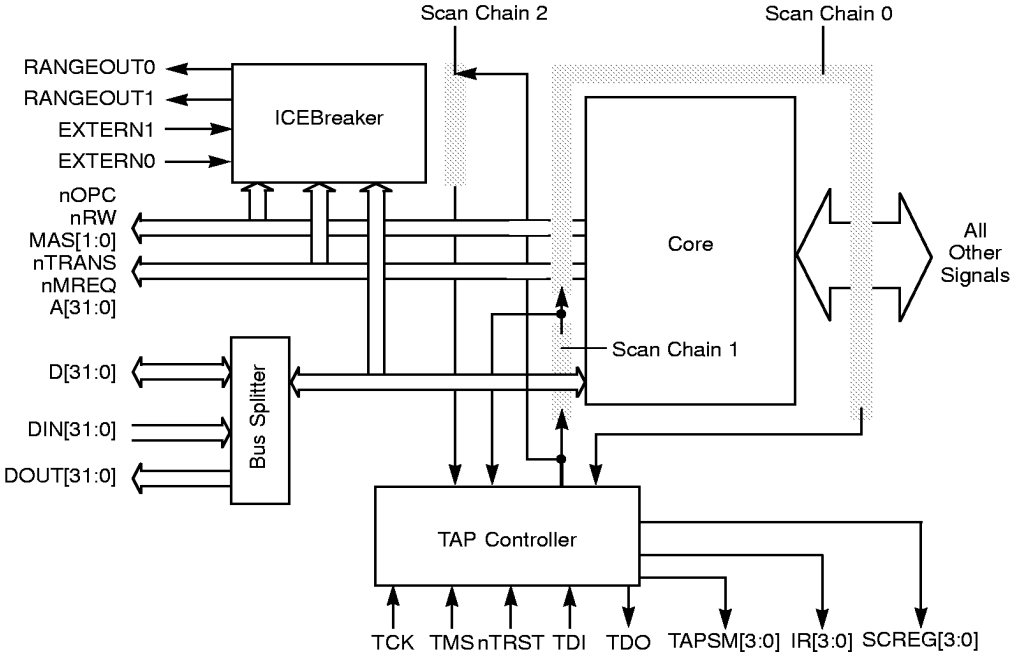
CW001004 ARM7TDMI Microprocessor Core

Preliminary Datasheet

The LSI Logic CW001004 core is an implementation of the Advanced RISC Machines ARM7TDMI 32-bit RISC microprocessor developed by Advanced RISC Machines. It was developed to meet the requirements of the LSI Logic CoreWare® methodology. Because it can execute both 32-bit and 16-bit instructions, it achieves code densities similar to many 16-bit CISC-based microcontrollers, but at substantially higher performance and lower power.

Figure 1 Core Block Diagram



The CW001004 core employs an innovative architectural strategy known as *THUMB*, which makes it ideally suited to high volume applications with memory restrictions and applications where code density is an issue. See “THUMB Architecture,” on page 7 for more information.

The ARM7TDMI core employs both the ARM and THUMB instruction sets which allows it to borrow from the wealth of development tools and third-party RTOS (real-time operating system) vendors that support the ARM architecture.

With its small size, high performance, low power requirements, and superior code density, the ARM7TDMI core is ideal for a wide variety of embedded applications.

The ARM architecture is based on Reduced Instruction Set Computer (RISC) principles, so the instruction set and related decode mechanism are much simpler than those of microprogrammed Complex Instruction Set Computers (CISC). This simplicity results in a high instruction throughput and impressive real-time interrupt response from a small and cost-effective chip.

The ARM memory interface has been designed to allow the performance potential to be realized without incurring high costs in the memory system. Speed critical control signals are pipelined to exploit the fast local access modes offered by industry-standard dynamic RAMs.

Features and Benefits

♦ 3.3 V, 0.25 micron (Leff) G10™ technology	♦ Access to full CoreWare library ♦ Low power and high density ♦ Small die size increases the area available for integration of other logic
♦ RTL design approach	♦ Simplified process migration ♦ Simplified CPU customization ♦ Complete and accurate timing model
♦ ARM 32-bit RISC execution engine delivers up to 55 MIPS	♦ Compatibility with a wide range of parts from other vendors ♦ Supported by an existing array of ARM development tools and RTOS
♦ Existing set of companion peripherals	♦ Simplifies system development
♦ Built-in code decompression (THUMB)	♦ Reduces total system memory requirements
♦ Includes full scan test structures	♦ Very high fault coverage for manufacturing test
♦ Includes ARM's ICEBreaker™ debugger	♦ Ideal for deeply embedded ASICs

Description

Figure 1 shows a block diagram of the CW001004 core. The CW001004 core consists of four major blocks: ICEBreaker, TAP Controller, Bus Splitter, and the Microprocessor (shown in more detail in Figure 2).

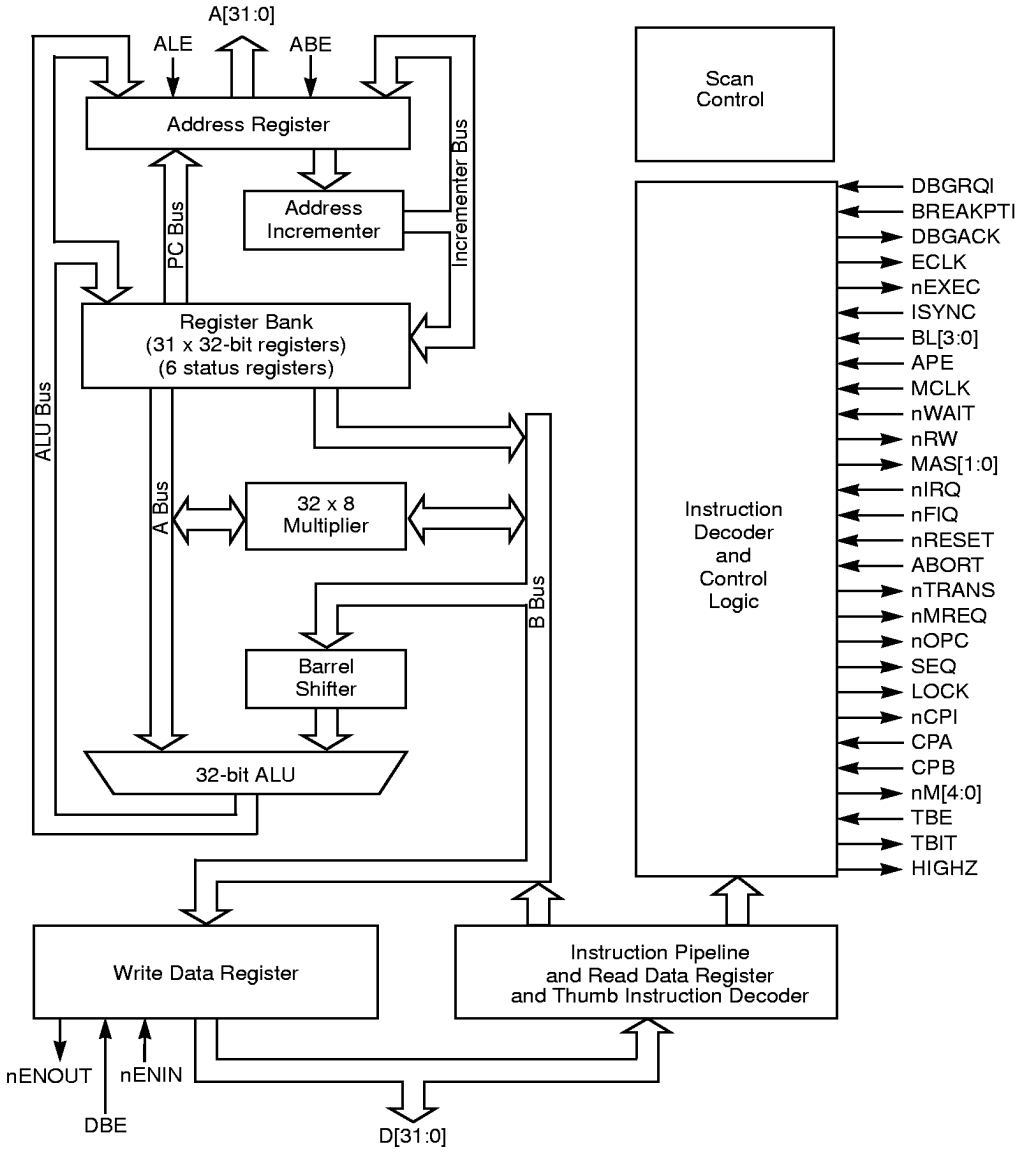
The ICEBreaker module provides integrated debugging support for the Internal Core module. It consists of two real-time watchpoint units, a control register, and a status register.

The TAP (Test Access Port) controller module controls three JTAG scan chains used for testing, debugging, and programming the ICEBreaker. A fourth scan chain is also provided for an external boundary chain around the pads of a packaged device.

The Bus Splitter is used to split the internal bidirectional data bus into three unidirectional buses for ASIC designs that prohibit bidirectional data buses.

The microprocessor module provides the main functionality of the microprocessor core. Figure 2 shows the microprocessor module in more detail.

Figure 2 Microprocessor Module Block Diagram



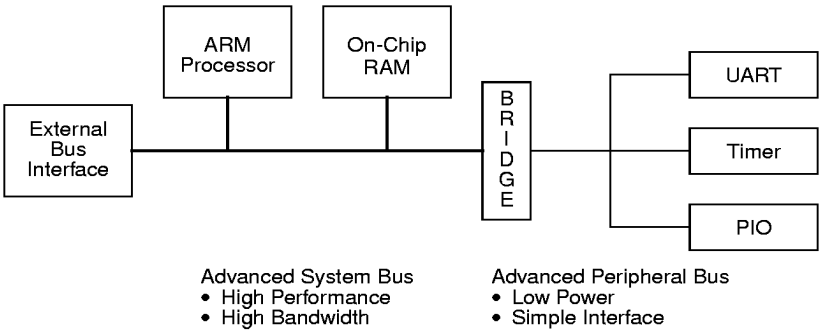
Pipelining

A three-stage pipeline is employed so that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory.

Building Blocks

To facilitate system design, LSI offers a complete library of peripherals implemented around the popular AMBA open bus standard. These peripherals can be used as-is, or modified to suit the specific application. Figure 3 shows a typical AMBA system that incorporates the ARM7TDMI core.

Figure 3. A Typical AMBA System



Debug and Full Scan

The ARM7TDMI core is implemented using full scan methodology for high fault coverage and contains the ARM ICEBreaker module for effective debugging, even in the most deeply embedded ASICs. As part of the LSI Logic's CoreWare Library, the ARM7TDMI core is supported by LSI Logic's ASIC design methodology, tools, and expert technical support.

THUMB Architecture

The CW001004 processor has two instruction sets:

- Standard 32-bit ARM set
- A 16-bit THUMB set

The THUMB set is a super-reduced instruction set. Its 16-bit instruction length allows it to approach twice the density of standard ARM code while retaining most of the ARM's performance advantage over a traditional 16-bit processor using 16-bit registers. This is possible because THUMB code operates on the same 32-bit register set as ARM code.

THUMB code is able to provide up to 65% of the code size of ARM and 160% of the performance of an equivalent ARM processor connected to a 16-bit memory system.

THUMB instructions operate with the standard ARM register configuration, allowing excellent interoperability between ARM and THUMB states. Each 16-bit THUMB instruction has a corresponding 32-bit ARM instruction with the same effect on the processor model.

The major advantage of a 32-bit (ARM) architecture over a 16-bit architecture is its ability to manipulate 32-bit integers with single instructions and to address a large address space efficiently. When processing 32-bit data, a 16-bit architecture will take at least two instructions to perform the same task as a single ARM instruction. However, not all the code in a program will process 32-bit data (for example, code that performs character string handling), and some instructions, like Branch instructions, do not process any data at all.

If a 16-bit architecture only has 16-bit instructions, and a 32-bit architecture only has 32-bit instructions, then overall the 16-bit architecture will have better code density and better than one half the performance of the 32-bit architecture. Clearly 32-bit performance comes at the cost of code density.

THUMB breaks this constraint by implementing a 16-bit instruction length on a 32-bit architecture, making the processing of 32-bit data efficient with a compact instruction coding. This provides far better performance than a 16-bit architecture, with better code density than a 32-bit architecture.

THUMB also has a major advantage over other 32-bit architectures with 16-bit instructions. This is the ability to switch back to full ARM code and execute at full speed. This enables critical loops for applications such as Fast interrupts and DSP algorithms to be coded using the full ARM instruction set and linked with THUMB code. The overhead of switching from THUMB code to ARM code is folded into subroutine entry time. Various portions of a system can be optimized for speed or for code density by switching between THUMB and ARM execution as appropriate.

Instruction Set Summary

The CW001004 has two instruction sets—a standard 32-bit ARM set, and a 16-bit THUMB set. Table 1 summarizes the ARM instruction set, and Table 2 summarizes the THUMB set.

Table 1. ARM Instruction Set (32-bit)

Op	Description
Branch Instructions	
B	Branch
BL	Branch with Link
BX	Branch and Exchange
Data Processing Instructions	
ADC	Add with Carry
ADD	Add
AND	Logical AND
BIC	Bit Clear
CMN	Compare Negated
CMP	Compare
EOR	Logical Exclusive OR
MOV	Move
MVN	Move Not
ORR	Logical (Inclusive) OR
RSB	Reverse Subtract
RSC	Reverse Subtract with Carry
SBC	Subtract with Carry
SUB	Subtract
TEQ	Test Equivalence
TST	Test
Multiply Instructions	
MLA	Multiply Accumulate
MUL	Multiply
MLAL	Multiply Accumulate Long
MULL	Multiply Long
Program Status Register (PSR) Transfer Instructions	
MRS	Move PSR Status/Flags to Register
MSR	Move register to PSR Status/Flags
Load and Store Instructions	
LDx	Load Multiple Registers, Byte, Word, Halfword
STx	Store Multiple Registers, Byte, Word, Halfword
Semaphore Instructions	
SWPx	Swap Word/Byte Between Register and Memory
Coprocessor Instructions	
CDP	Coprocessor Data Processing
LDC	Load Coprocessor from Memory
MCR	Move to Coprocessor Register from ARM
MRC	Move to ARM Register from Coprocessor
STC	Store Coprocessor Register to Memory
Interrupt Instructions	
SWI	Software Interrupt

Table 2. Thumb Instruction Set (16-bit)

Op	Description
Branch Instructions	
B	Branch
BL	Branch with Link
BX	Branch and Exchange
Data Processing Instructions	
ADC	Add with Carry
ADD	Add
AND	Logical AND
ASR	Arithmetic Shift Right
BIC	Bit Clear
CMN	Compare Negative
CMP	Compare
EOR	Exclusive OR
LSL	Logical Shift Left
LSR	Logical Shift Right
MOV	Move
MVN	Move NOT
NEG	Negate
ORR	Logical OR
ROR	Rotate Right
SBC	Subtract with Carry
SUB	Subtract
TST	Test Bits
Multiply Instructions	
MUL	Multiply
Load and Store Instructions	
LDx	Load Multiple Registers, Byte, Word, Halfword
POP	Pop Registers
PUSH	Push Registers
STx	Store Multiple Registers, Byte, Word, Halfword
Interrupt Instructions	
SWI	Software Interrupt

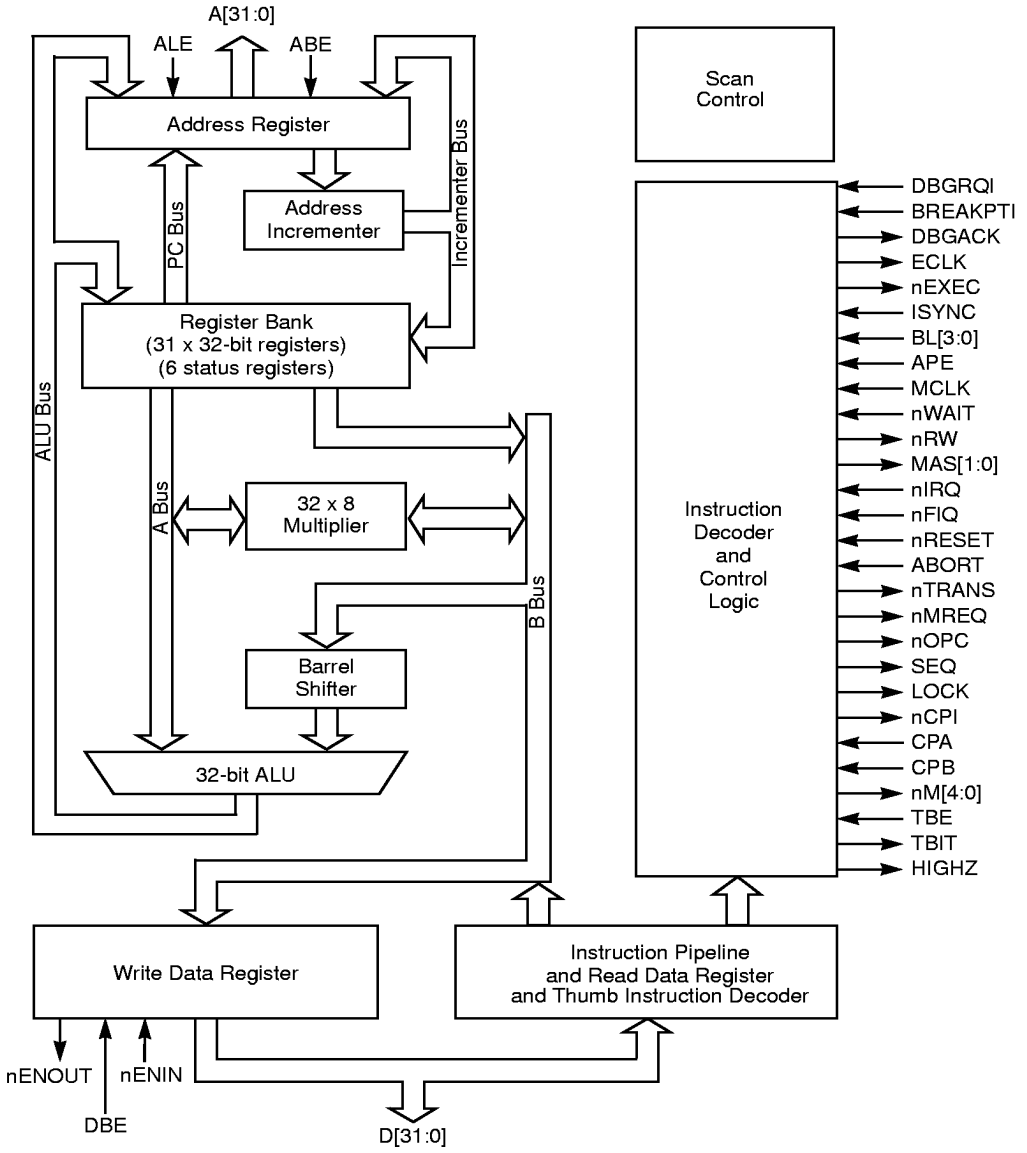
Signal Descriptions

The CW001004 signals are shown in Figure 3, and are listed by functional group:

- Clocks
- Interrupts
- Bus Controls
- Debug
- Scan Test
- Boundary Scan
- Boundary Scan Control
- Processor Interface
- Memory Interface
- Memory Management Interface
- Coprocessor Interface

In the descriptions that follow, the verb *assert* means to drive TRUE or active. The verb *deassert* means to drive FALSE or inactive.

Figure 4 ARM7TDMI Logic Diagram



Clocks

MCLK — **Memory Clock Input** — **Input**
This clock times all CW001004 memory accesses and internal operations.

nWAIT — **Not Wait** — **Input**
Deasserting this signal makes the core wait for an integer number of MCLK cycles.

ECLK — **External Clock Output** — **Output**
In normal operation, this is MCLK exported from the core. When the core is being debugged, this is DCLK.

Interrupts

nFIQ — **Not Fast Interrupt Request** — **Input**
An interrupt request that causes the processor to be interrupted if taken LOW.

nIRQ — **Not Interrupt Request** — **Input**
This is the same as nFIQ, but with lower priority.

ISYNC — **Synchronous Interrupts** — **Input**
When LOW, this signal indicates that interrupt request inputs are to be synchronized by the ARM core.

Bus Controls

nRESET — **Not Reset** — **Input**
A LOW level causes the instruction being executed to terminate abnormally. When HIGH for at least one clock cycle, the processor restarts from address 0.

BUSEN — **Data Bus Configuration** — **Input**
Determines whether the bidirectional data bus or the unidirectional data busses are to be used.

HIGHZ — **High Z Instruction** — **Output**
This signal denotes that the HIGHZ instruction has been loaded into the TAP controller.

BIGEND — **Big Endian Configuration** — **Input**
When this signal is HIGH the processor treats bytes in memory as being in Big Endian format.

nENIN **NOT Enable Input** **Input**
Used with nENOUT to control the data bus during write cycles.

nENOUT **Not Enable Output** **Output**
During a data write cycle, this signal is driven LOW during phase 1, and remains LOW for the entire cycle.

nENOUTI **Not Enable Output** **Output**
Used to aid arbitration in shared bus systems.

ABE **Address Bus Enable** **Input**
When LOW, puts the address bus drivers into a high impedance state.

APE **Address Pipeline Enable** **Input**
When HIGH, this signal enables the address timing pipeline.

ALE **Address Latch Enable** **Input**
Used to control transparent latches on the address outputs.

DBE **Data Bus Enable** **Input**
When driven LOW, puts the data bus into the high impedance state.

TBE **Test Bus Enable** **Input**
When LOW, puts the data bus, the address bus, LOCK, MAS[1:0], nRW, nTRANS, and nOPC into a high impedance state.

BUSDIS **Bus Disable** **Output**
Used to disable external logic driving onto the bidirectional data bus during scan testing.

ECAPCLK **Extest Capture Clock** **Output**
Removes the need for external logic to enable the internal 3-state bus during scan testing.

Debug

DBGRQ **Debug Request** **Input**
When HIGH causes the core to enter debug state after executing the current instruction.

BREAKPT **Breakpoint** **Input**
When HIGH, causes the current memory access to be breakpointed.

DBGACK **Debug Acknowledge** **Output**
When HIGH, indicates that the core is in debug state.

nEXEC **Not Executed** **Output**
When HIGH, indicates that the instruction in the execution unit is not being executed.

EXTERN0 **External Input 0** **Input**
Input to ICEBreaker logic that allows breakpoints and/or watchpoints to be dependent on an external condition.

EXTERN1 **External Input 1** **Input**
Input to ICEBreaker logic that allows breakpoints and/or watchpoints to be dependent on an external condition.

DBGEN **Debug Enable** **Input**
Disables the debug features of the core.

RANGEOUT0 **ICEbreaker Rangeout 0** **Output**
Indicates that the ICEBreaker watchpoint register 0 has matched the conditions currently present on the address, data and control busses.

RANGEOUT1 **ICEbreaker Rangeout 1** **Output**
This signal is the same as RANGEOUT0 but corresponds to the ICEBreaker watchpoint register 1.

DBGRQI **Internal Debug Request** **Output**
Represents the debug request signal that is presented to the processor.

COMMRX **Communications Channel Receive** **Output**
When HIGH, denotes that the communications channel receive buffer is empty.

COMMTX **Communications Channel Transmit** **Output**
When HIGH, denotes that the communications channel transmit buffer is empty.

Scan Test

FULLSCAN **Master Scan Mode Select** **Input**
Enables full scan input.

RAMTEST **Ramtest Scan Mode Select** **Input**
Places the core in Ramtest Mode, if FULLSCAN is asserted.

RAMTEST_IN
Ramtest Scan Chain Input **Input**
Scan input for the core memory scan chain in Ramtest.

RAMTEST_OUT
Ramtest Scan Chain Output **Output**
Scan output for the core memory scan chain in Ramtest mode.

SCAN_EN **Global Scan Enable** **Input**
Enables serial loading of the scan registers through the scan chain in Production Test or Ramtest mode.

SCAN_IN **Full Scan Chain Input** **Input**
Scan input for core memory scan chain in Production Test mode.

SCAN_OUT **Full Scan Chain Output** **Output**
Scan output for the core memory scan chain in Production Test mode.

WENCTEST **Ramtest write enable** **Input**
Controls core memory writes in Ramtest mode.

Boundary Scan

TCK **Test Clock** **Input**
The clock used for test operations.

TCK1 **TCK, Phase 1** **Output**
This clock represents phase 1 of TCK.

TCK2 **TCK, Phase 2** **Output**
This clock represents phase 2 of TCK.

TMS **Test Mode Select** **Input**
Selects the test mode.

TDI	**Test Data Input** This signal is for test data input.	**Input**
TDO	**Test Data Output** Output from the boundary scan logic.	**Output**
nTRST	**Not Test Reset** Active LOW reset signal for the boundary scan logic.	**Input**
TAPSM[3:0]	**TAP Controller State Machine** Shows the current state of the TAP controller state machine.	**Output**
IR[3:0]	**TAP Controller Instruction Register** Shows the current instruction loaded into the TAP controller instruction register.	**Output**
nTDOEN	**Not TDO Enable** When LOW, this signal denotes that serial data is being driven out on the TDO output.	**Output**
SCREG[3:0]	**Scan Chain Register** Shows the ID number of the scan chain currently selected by the TAP controller.	**Output**

Boundary Scan Control

DRIVEBS	**Boundary Scan Cell Enable** Controls the multiplexers in the scan cells of an external boundary scan chain.	**Output**
ECAPCLKBS	**Extest Capture Clock for Boundary Scan** Clock used to capture the core outputs during EXTEST.	**Output**
ICAPCLKB	**SIntest Capture Clock** Clock used to capture the core outputs during INTEST.	**Output**
nHIGHZ	**Not HIGHZ** Places the scan cells of a scan chain in the high impedance state.	**Output**
PCLKBS	**Boundary Scan Update Clock** Used by an external boundary scan chain as the update clock.	**Output**

RSTCLKBS **Boundary Scan Reset Clock** **Output**
Denotes that either the TAP controller state machine is in the RESET state or that nTRST has been asserted.

SDINBS **Boundary Scan Serial Input Data** **Output**
The serial data to be applied to an external scan chain.

SDOUTBS **Boundary Scan Serial Output Data** **Input**
The serial data out of the boundary scan chain.

SHCLKBS **Boundary Scan Shift Clock, Phase 1** **Output**
Used to clock the master element of the scan cells.

SHCLK2BS **Boundary Scan Shift Clock, Phase 2** **Output**
Used to clock the slave element of the scan cells.

Processor Interface

nM[4:0] **Not Processor Mode** **Output**
Output signals that are the inverse of the internal status bits indicating the processor operation mode.

TBIT **Thumb Instruction Set Enable** **Output**
When HIGH, denotes that the processor is executing the THUMB instruction set. When LOW, the processor is executing the ARM instruction set.

Memory Interface

A[31:0] **Addresses** **Output**
This is the processor address bus.

D[31:0] **Data Bus** **Bidirectional**
Bidirectional bus for data transfers between the processor and external memory.

DOUT[31:0] **Data Output Bus** **Output**
This is the data out bus, used to transfer data from the processor to the memory system.

DIN[31:0] **Data Input Bus** **Input**
The input data bus used to transfer instructions and data between the processor and memory.

nMREQ	**Not Memory Request**	**Output**
	When LOW, this signal indicates that the processor requires memory access during the following cycle.	
SEQ	**Sequential Address**	**Output**
	Becomes HIGH when the address of the next memory cycle is related to that of the last memory access.	
nRW	**Not Read/Write**	**Output**
	When HIGH, indicates a processor write cycle; when LOW, a read cycle.	
MAS[1:0]	**Memory Access Size**	**Output**
	Indicates to the external memory system whether a word, halfword, or byte length transfer is required.	
BL[3:0]	**Byte Latch Control**	**Input**
	These signals control when data and instructions are latched from the external data bus.	
LOCK	**Locked Operation**	**Output**
	When LOCK is HIGH, the processor is performing a “locked” memory access.	

Memory Management Interface

nTRANS	**Not Memory Translate**	**Output**
	When this signal is LOW it indicates that the processor is in user mode.	
ABORT	**Memory Abort**	**Input**
	Allows the memory system to tell the processor that a requested access is not allowed.	
nOPC	**Not Op-code Fetch**	**Output**
	When LOW, indicates that the processor is fetching an instruction from memory.	

Coprocessor Interface

nCPI	**Not Coprocessor Instruction**	**Output**
	This output is taken LOW when the core starts to execute a coprocessor instruction.	

CPA **Coprocessor Absent** **Input**
A coprocessor that is capable of performing the operation that CW001004 is requesting should take CPA LOW immediately.

CPB **Coprocessor Busy** **Input**
A coprocessor that is capable of performing the requested operation, but cannot commit to starting it immediately, should indicate this by driving CPB HIGH.

Specifications

This section specifies the CW001004 architecture's electrical and mechanical characteristics. The timing parameters given here are preliminary data and subject to change.

AC Timing

Output load is 0.24 pF.

In the two tables that follow the letters after the signal name refer to: rising edge (r) and falling edge (f).

Values are given for two operating conditions:

- Table 3: Tj = 70 °C, 3.135 V (3.3 V – 5%)
- Table 4: Tj = 85 °C, 2.7 V

Table 3. AC Parameters - Tj = 70 °C, 3.135 V (3.3 V – 5%)

Symbol	Parameter	Min	Max
T_{mck}	MCLK Cycle Time	16	
T_{mckl}	MCLK LOW Time	6.2	
T_{mckh}	MCLK HIGH Time	6.0	
T_{ws}	nWAIT Setup to MCLKr	0.5	
T_{wh}	nWAIT Hold from CKf	1.6	
T_{ale}	Address Latch Open		3.4
T_{aleh}	Address Latch Hold Time	2.2	

Table 3. AC Parameters - Tj = 70 °C, 3.135 V (3.3 V – 5%) (Cont.)

Symbol	Parameter	Min	Max
T_{ald}	Address Latch Time		4.5
T_{addr}	MCLKr to Address Valid		9.4
T_{ah}	Address Hold Time from Mclkr	6.5	
T_{abe}	Address Bus Enable Time		2.3
T_{abz}	Address Bus Disable Time		2.5
T_{aph}	APE Hold Time from MCLKr	2.5	
T_{aps}	APE Setup Time to MCLKf	1.7	
T_{ape}	MCLKf to Address Valid		5.5
T_{apeh}	Address Group Hold Time from MCLKf	4.4	
T_{dout}	MCLKf to D[31:0] Valid		8.1
T_{doh}	D[31:0] Out Hold from MCLKf	4.9	
T_{dis}	D[31:0] In Setup Time to MCLKf	0.9	
T_{dih}	D[31:0] In Hold Time from MCLKf	0.6	
T_{doutu}	MCLKf to DOUT[31:0] Valid		5.5
T_{dohu}	DOUT[31:0] Hold Time from MCLKf	4.1	
T_{disu}	DIN[31:0] Setup Time to MCLKf	1.5	
T_{dihu}	DIN Hold Time to MCLKf	3.1	
T_{nen}	MCLKf to nENOUT Valid		7.2
T_{nenh}	nENOUT Hold Time from MCLKf	3.2	
T_{bylh}	BL[3:0] Hold Time from MCLKf	0	
T_{byls}	BL[3:0] Setup to from MCLKr	1.0	
T_{dbe}	Data Bus Enable Time from DBEr		3.2
T_{dbz}	Data Bus Disable Time from DBEf		3.2
T_{dbnen}	DBE to nENOUT Valid		2.0
T_{tbz}	Address and Data Bus Disable Time from TBEf		2.4

Table 3. AC Parameters - Tj = 70 °C, 3.135 V (3.3 V – 5%) (Cont.)

Symbol	Parameter	Min	Max
T_{tbe}	Address and Data Bus Enable Time from TBEr		2.6
T_{rwd}	MCLKr to nRW Valid		5.6
T_{rwh}	nRW Hold Time from MCLKr	4.5	
T_{msd}	MCLKf to nMREQ & SEQ Valid		14.7
T_{msh}	nMREQ and SEQ Hold Time from MCLKf	4.7	
T_{bld}	MCLKr to MAS[1:0] and LOCK		10.1
T_{blh}	MAS[1:0] and LOCK Hold from MCLKr	4.3	
T_{mdd}	MCLKr to nTRANS, nM[4:0], and TBIT Valid		6.6
T_{mdh}	nTRANS and nM[4:0] Hold Time from MCLKr	3.6	
T_{opcd}	MCLKr to nOPC Valid		5.9
T_{opch}	nOPC Hold Time from MCLKr	4.1	
T_{cps}	CPA, CPB Setup to MCLKr	0	
T_{cph}	CPA,CPB Hold Time from MCLKr	4.4	
T_{cpms}	CPA, CPB to nMREQ, SEQ		9.1
T_{cpi}	MCLKf to nCPI Valid		10.5
T_{cpih}	nCPI Hold Time from MCLKf	5.1	
T_{cts}	Config Setup Time	2.2	
T_{cth}	Config Hold Time	3.5	
T_{abts}	ABORT Setup Time to MCLKf	0	
T_{abth}	ABORT Hold Time from MCLKf	3.8	
T_{is}	Asynchronous Interrupt Setup Time to MCLKf for Guaranteed Recognition (ISYNC = 0)	0	
T_{im}	Asynchronous Interrupt Guaranteed Nonrecognition Time (ISYNC = 0)		4.4
T_{sis}	Synchronous nFIQ, nIRQ Setup to MCLKf (ISYNC = 1)	0	
T_{sih}	Synchronous nFIQ, nIRQ Hold from MCLKf (ISYNC = 1)	4.0	

Table 3. AC Parameters - Tj = 70 °C, 3.135 V (3.3 V – 5%) (Cont.)

Symbol	Parameter	Min	Max
T_{rs}	Reset Setup Time to MCLKr for Guaranteed Recognition	0	
T_{rm}	Reset Guaranteed Nonrecognition Time	6.9	
T_{exd}	MCLKf to nEXEC Valid		12.3
T_{exh}	nEXEC Hold Time from MCLKf	6.2	
T_{brks}	Setup Time of BREAKPT to MCLKr	7.6	
T_{brkh}	Hold Time of BREAKPT from MCLKr		0
T_{bcems}	BREAKPT to nCPI, nEXEC, nMREQ, SEQ Delay		11.2
T_{dbgd}	MCLKr to DBGACK Valid		7.6
T_{dbgh}	DGBACK Hold Time from MCLKr	6.6	
T_{rqs}	DBGRQ Setup Time to MCLKr for Guaranteed Recognition	0	
T_{rqh}	DBGRQ Guaranteed Nonrecognition time	5.0	
T_{cdel}	MCLK to ECLK Delay		3.9
T_{ctdel}	TCK to ECLK Delay		4.0
T_{exts}	EXTERN[1:0] Setup Time to MCLKf	0	
T_{exth}	EXTERN[1:0] Hold Time from MCLKf	4.83	
T_{rg}	MCLKf to RANGEOUT0, RANGEOUT1 Valid		8.2
T_{rgh}	RANGEOUT0, RANGEOUT1 Hold Time from MCLKf	6.2	
T_{dbgrq}	DBGRQ to DBGRQI Valid		2.5
T_{rstd}	nRESETf to D[], DBGACK, nCPI, nENOUT, nEXEC, nMREQ, SEQ Valid		9.4
T_{commd}	MCLKr to COMMRX, COMMTX Valid		6.6
T_{trstd}	nTRSTf to Every Output Valid		15.8
T_{rstl}	nRESET LOW for Guaranteed Reset	2 MCLK cycles	

Table 4. AC Parameters - Tj = 85 °C, 2.7 V

Symbol	Parameter	Min	Max
T_{mck}	MCLK Cycle Time	18.5	
T_{mckl}	MCLK LOW Time	7.1	
T_{mckh}	MCLK HIGH Time	7.1	
T_{ws}	nWAIT Setup to MCLKr	0.8	
T_{wh}	nWAIT Hold from CKf	1.4	
T_{ale}	Address Latch Open		4.0
T_{aleh}	Address Latch Hold Time	2.3	
T_{ald}	Address Latch Time		5.3
T_{addr}	MCLKr to Address Valid		13.0
T_{ah}	Address Hold Time from MCLKr	7.5	
T_{abe}	Address Bus Enable Time		2.5
T_{abz}	Address Bus Disable Time		2.7
T_{aph}	APE Hold Time from MCLKr	2.8	
T_{aps}	APE Setup Time to MCLKf	2.0	
T_{ape}	MCLKf to Address Valid		6.5
T_{apeh}	Address Group Hold Time from MCLKf	5.0	
T_{dout}	MCLKf to D[31:0] Valid		9.72
T_{doh}	D[31:0] Out Hold from MCLKf	5.6	
T_{dis}	D[31:0] In Setup Time to MCLKf	1.3	
T_{dih}	D[31:0] In Hold Time from MCLKf	1.4	
T_{doutu}	MCLKf to DOUT[31:0] Valid		6.5
T_{dohu}	DOUT[31:0] Hold Time from MCLKf	4.7	
T_{disu}	DIN[31:0] Setup Time to MCLKf	1.8	
T_{dihu}	DIN[hold Time to MCLKf	3.6	
T_{nen}	MCLKf to nENOUT Valid		8.7

Table 4. AC Parameters - Tj = 85 °C, 2.7 V (Cont.)

Symbol	Parameter	Min	Max
T_{nenh}	nENOUT Hold Time from MCLKf	3.7	
T_{bylh}	BL[3:0] Hold Time from MCLKf	0	
T_{byls}	BL[3:0] Setup to from MCLKr	1.5	
T_{dbe}	Data Bus Enable Time from DBEr		3.9
T_{dbz}	Data Bus Disable Time from DBEf		3.9
T_{dbnen}	DBE to nENOUT Valid		2.4
T_{tbz}	Address and Data Bus Disable Time from TBEf		2.8
T_{tbe}	Address and Data Bus Enable Time from TBEr		2.9
T_{rwd}	MCLKr to nRW Valid		6.5
T_{rwh}	nRW Hold Time from MCLKr	5.2	
T_{msd}	MCLKf to nMREQ and SEQ Valid		17.8
T_{msh}	nMREQ and SEQ Hold Time from MCLKf	5.4	
T_{bld}	MCLKr to MAS[1:0] and LOCK		12.0
T_{blh}	MAS[1:0] and LOCK Hold from MCLKr	4.5.	
T_{mdd}	MCLKr to nTRANS, nM[4:0], and TBIT Valid		7.8
T_{mdh}	nTRANS and nM[4:0] Hold Time from MCLKr	4.1	
T_{opcd}	MCLKr to nOPC Valid		6.9
T_{opch}	nOPC Hold Time from MCLKr	4.4	
T_{cps}	CPA, CPB Setup to MCLKr	0	
T_{cph}	CPA,CPB Hold Time from MCLKr	5.0	
T_{cpms}	CPA, CPB to nMREQ, SEQ		10.9
T_{cpi}	MCLKf to nCPI Valid		11.4
T_{cpih}	nCPI Hold Time from MCLKf	4.4	
T_{cts}	Config Setup Time	2.0	
T_{cth}	Config Hold Time	3.0	

Table 4. AC Parameters - Tj = 85 °C, 2.7 V (Cont.)

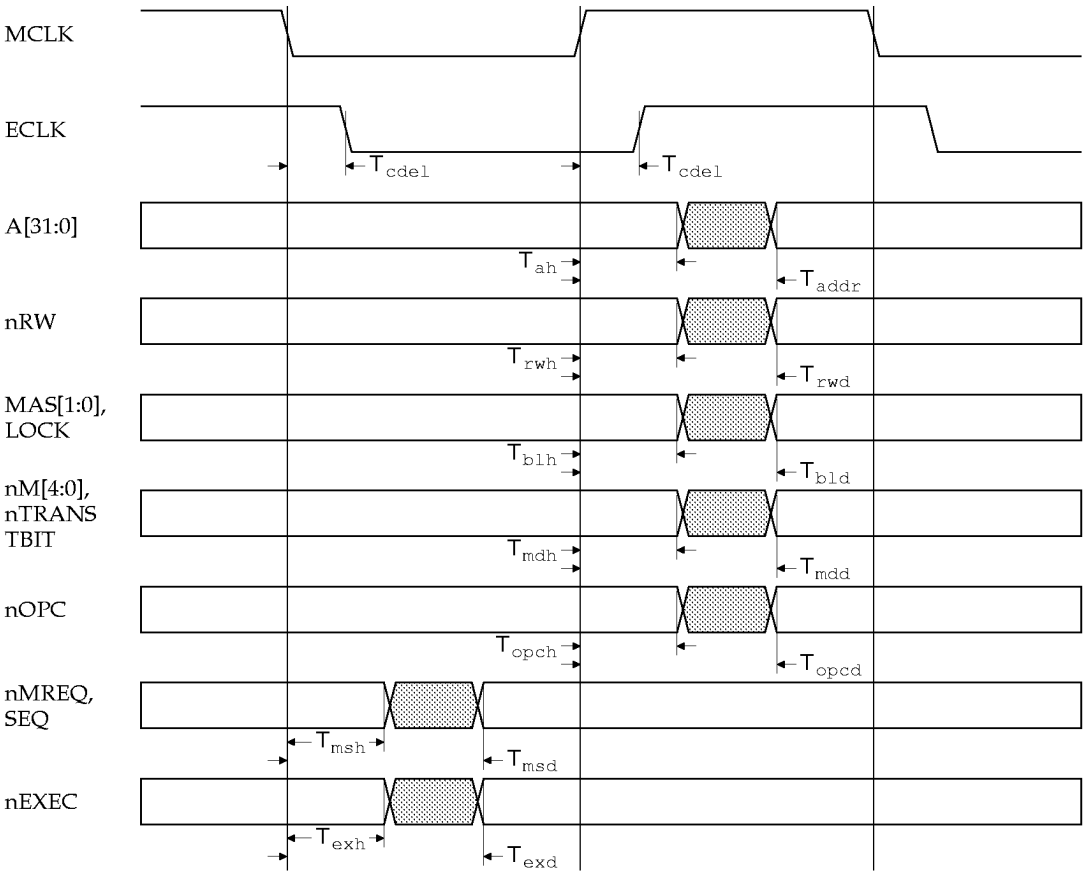
Symbol	Parameter	Min	Max
T_{abts}	ABORT Setup Time to MCLKf	0	
T_{abth}	ABORT Hold Time from MCLKf	3.3	
T_{is}	Asynchronous Interrupt Setup Time to MCLKf for Guaranteed Recognition (ISYNC = 0)	0	
T_{im}	Asynchronous Interrupt Guaranteed Nonrecognition Time (ISYNC = 0)		3.7
T_{sis}	Synchronous nFIQ, nIRQ Setup to MCLKf (ISYNC = 1)	0	
T_{sih}	Synchronous nFIQ, nIRQ Hold from MCLKf (ISYNC = 1)	3.6	
T_{rs}	Reset Setup Time to MCLKr for Guaranteed Recognition	0	
T_{rm}	Reset Guaranteed Nonrecognition Time	5.9	
T_{exd}	MCLKf to nEXEC Valid		10.7
T_{exh}	nEXEC Hold Time from MCLKf	4.8	
T_{brks}	Setup Time of BREAKPT to MCLKr	6.6	
T_{brkh}	Hold Time of BREAKPT from MCLKr		0
T_{bcems}	BREAKPT to nCPI, nEXEC, nMREQ, SEQ Delay		9.7
T_{dbgd}	MCLKr to DBGACK Valid		6.6
T_{dbgh}	DGBACK Hold Time from MCLKr	5.5	
T_{rqs}	DBGRQ Setup Time to MCLKr for Guaranteed Recognition	0	
T_{rqh}	DBGRQ Guaranteed Nonrecognition Time	4.3	
T_{cdel}	MCLK to ECLK Delay		3.4
T_{ctdel}	TCK to ECLK Delay		3.5
T_{exts}	EXTERN[1:0] Setup Time to MCLKf	0	
T_{exth}	EXTERN[1:0] Hold Time from MCLKf	4.2	
T_{rg}	MCLKf to RANGEOUT0, RANGEOUT1 Valid		7.1

Table 4. AC Parameters - Tj = 85 °C, 2.7 V (Cont.)

Symbol	Parameter	Min	Max
T_{rgh}	RANGEOUT0, RANGEOUT1 Hold Time from MCLKf	5.4	
T_{dbgrq}	DBGRQ to DBGRQI Valid		2.2
T_{rstd}	nRESETf to D[], DBGACK, nCPI, nENOUT, nEXEC, nMREQ, SEQ Valid		8.1
T_{commd}	MCLKr to COMMRX, COMMTX Valid		5.9
T_{trstd}	nTRSTf to Every Output Valid		13.6
T_{rstl}	nRESET LOW for Guaranteed Reset	2 MCLK cycles	

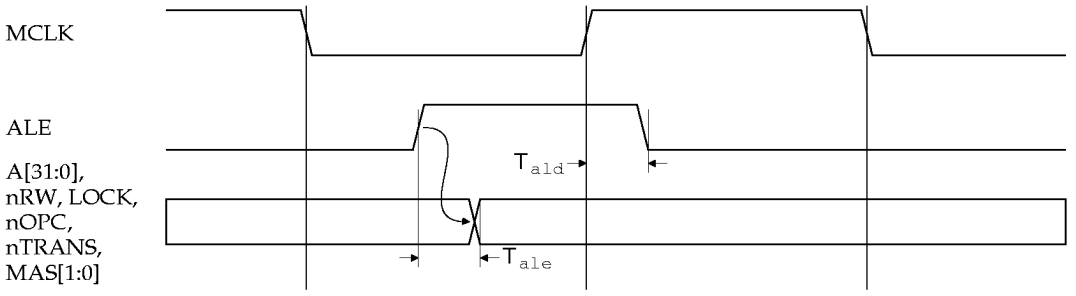
Timing Diagrams

Figure 5 General Timings



Note: nWAIT, APE, ALE and ABE are all HIGH during the cycle shown. T_{odel} is the delay (on either edge) from MCLK changing to ECLK changing.

Figure 6 ALE Address Control



Notes:
1. T_{ald} is the time by which ALE must be driven LOW in order to latch the current address in phase
2. If ALE is driven low after T_{ald}, then a new address will be latched.

Figure 7 APE Address Control

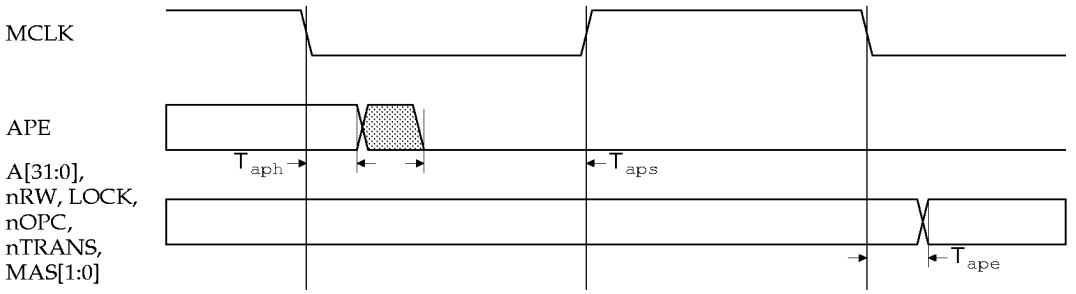


Figure 8 ABE Address Control

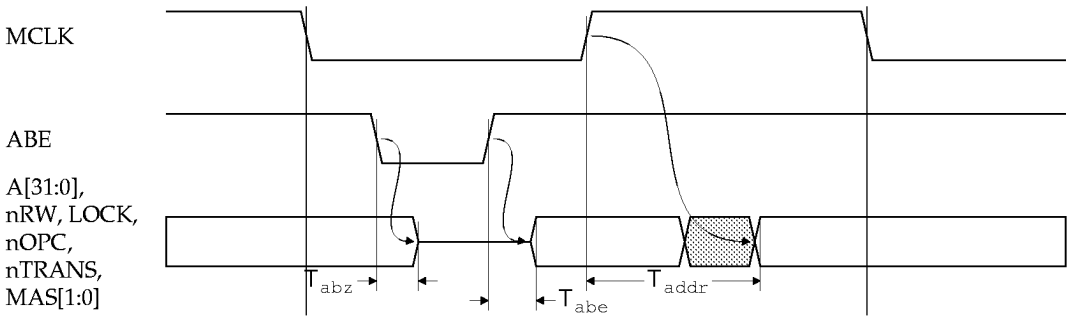
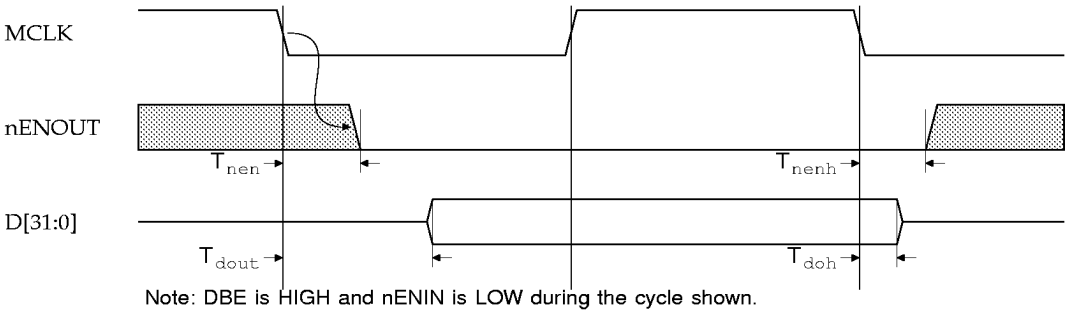
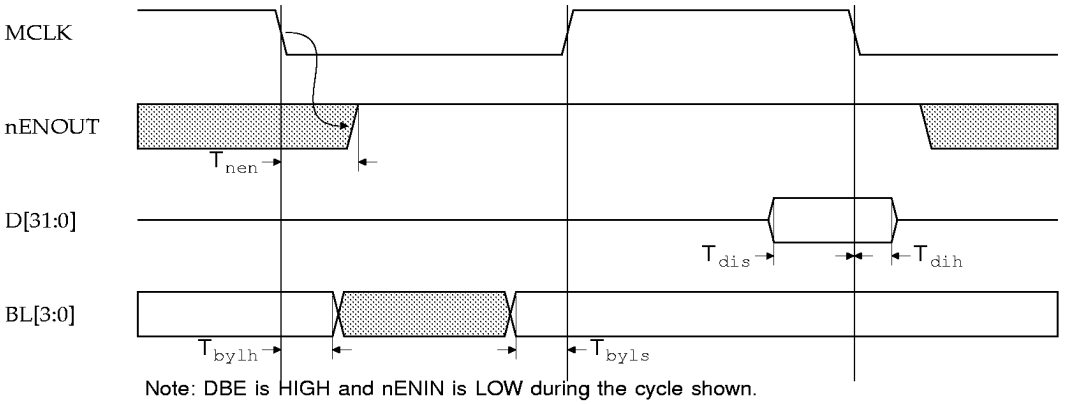


Figure 9 Bidirectional Data Write Cycle



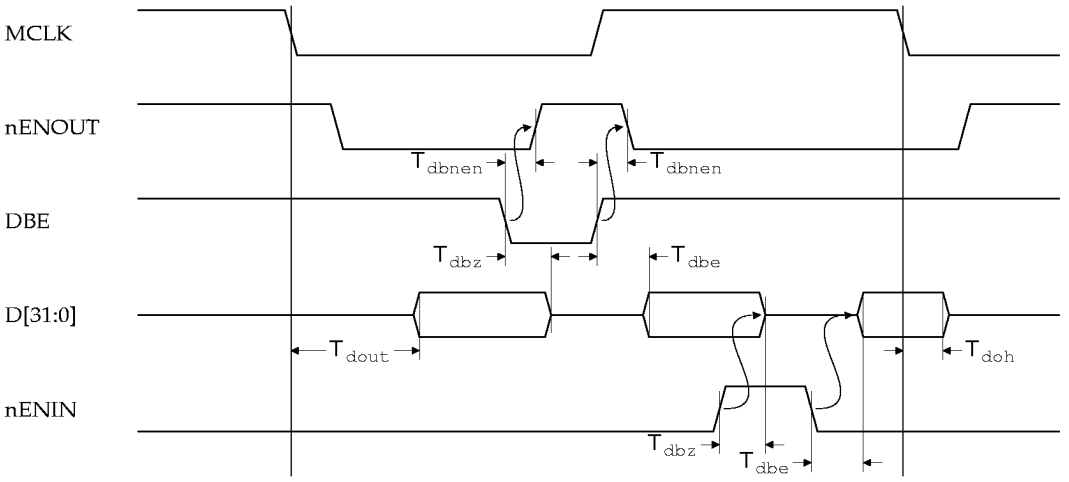
Note: DBE is HIGH and nENIN is LOW during the cycle shown.

Figure 10 Bidirectional Data Read Cycle



Note: DBE is HIGH and nENIN is LOW during the cycle shown.

Figure 11 Data Bus Control



Notes:
1. The cycle shown is a data write cycle since nENOUT was driven LOW during phase.
2. Here, DBE has first been used to modify the behavior of the data bus, and then nENIN.

Figure 12 Output 3-State Time

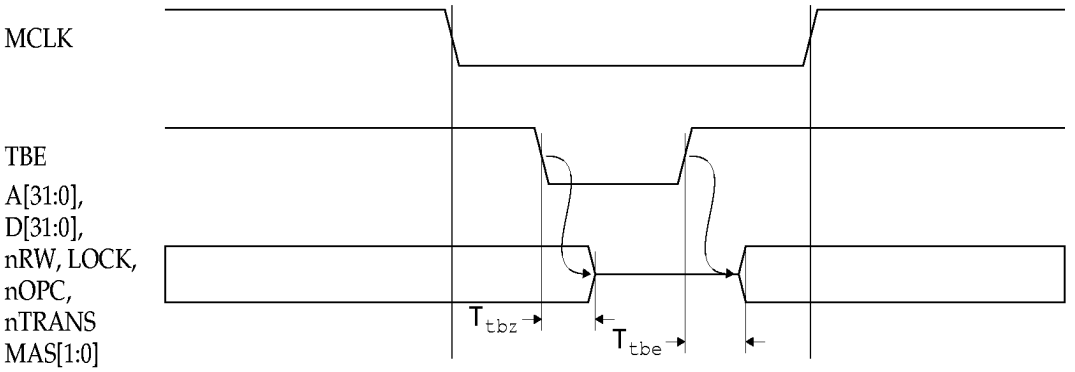


Figure 13 Unidirectional Data Write Cycle

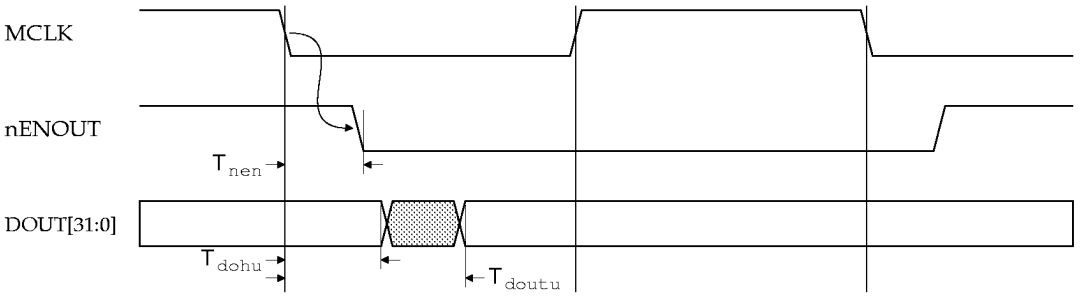


Figure 14 Unidirectional Data Read Cycle

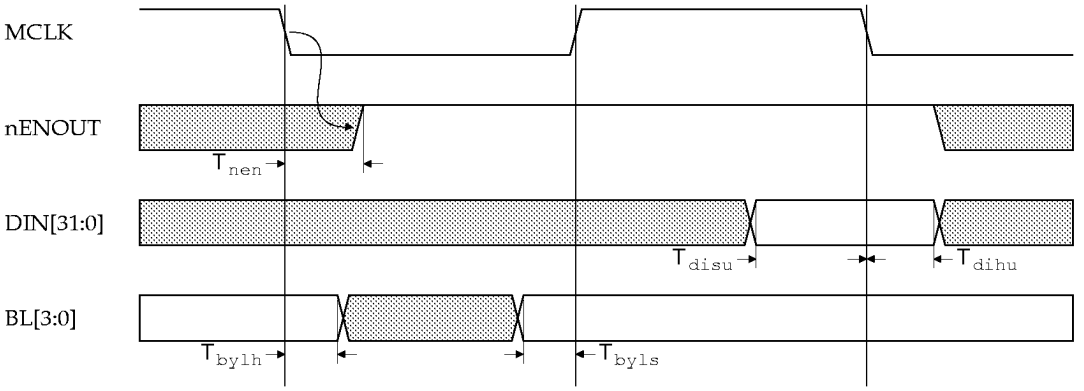


Figure 15 Configuration Pin Timing

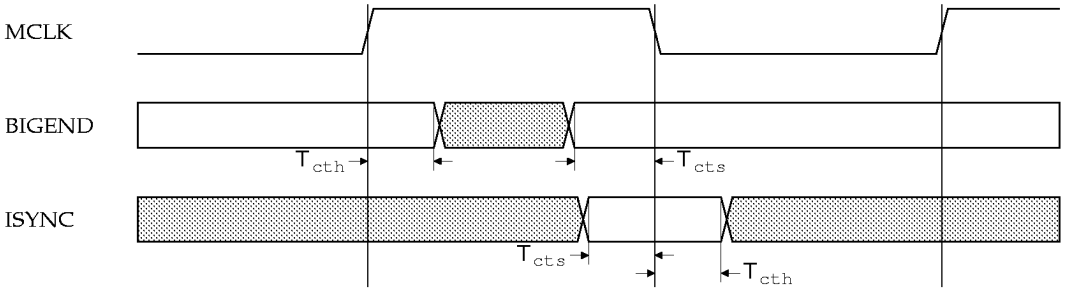
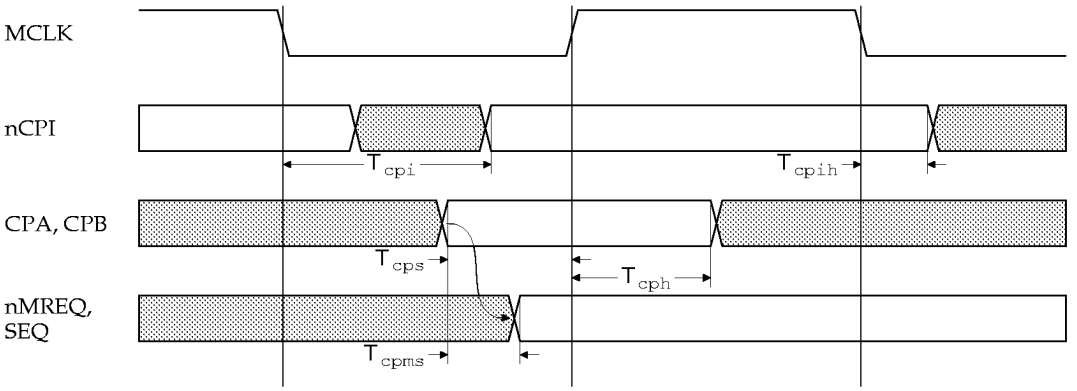
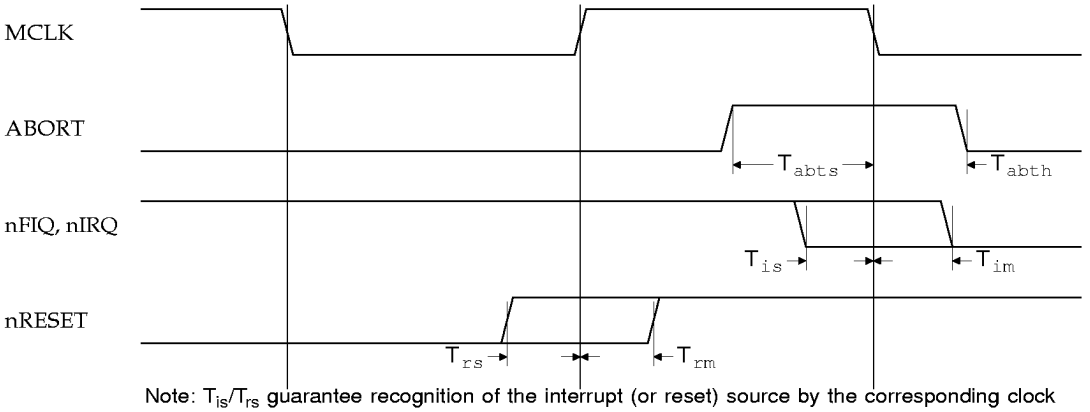


Figure 16 Coprocessor Timing



Note: Normally, nMREQ and SEQ become valid T_{msd} after the falling edge of MCLK. In this cycle the ARM has been busy-waiting, waiting for a coprocessor to complete the instruction. If CPA and CPB change during phase 1, the timing of nMREQ and SEQ will depend on T_{cpms}. Most systems should be able to generate CPA and CPB during the previous phase 2, and so the timing of nMREQ and SEQ will always be T_{msd}.

Figure 17 Exception Timing



Note: T_{is}/T_{rs} guarantee recognition of the interrupt (or reset) source by the corresponding clock edge. T_{im}/T_{rm} guarantee nonrecognition by that clock edge. These inputs may be applied fully asynchronously where the exact cycle of recognition is unimportant.

Figure 18 Debug Timing

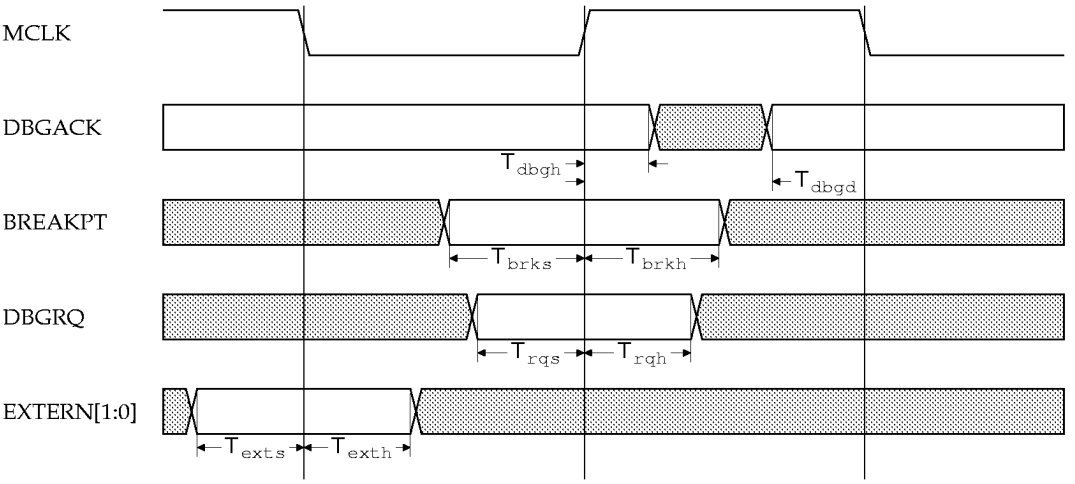
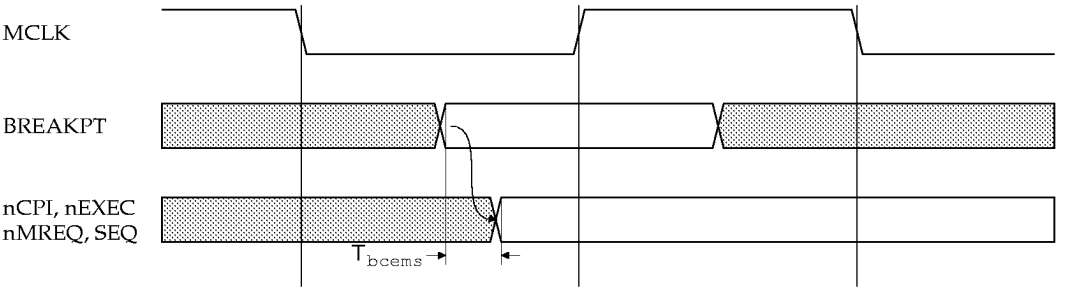


Figure 19 Breakpoint Timing



Note: BREAKPT changing in the LOW phase of MCLK to signal a watchpointed store can affect nCPI, nEXEC, nMREQ, and SEQ in the LOW phase of MCLK.

Figure 20 TCK-ECLK Relationship

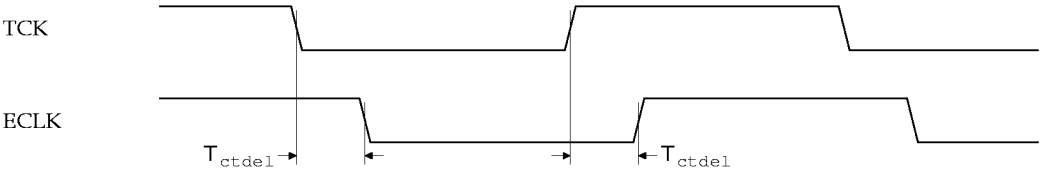
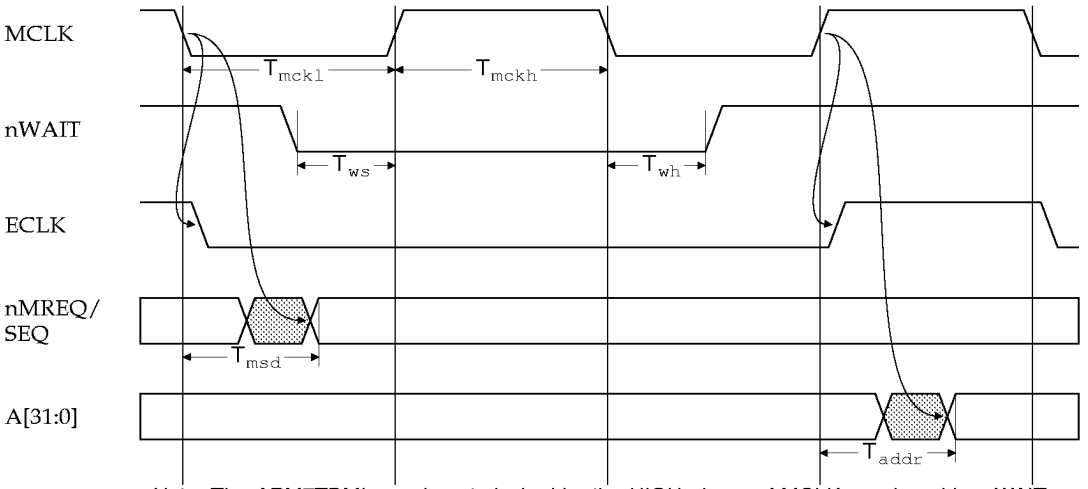


Figure 21 MCLK Timing



Note: The ARM7TDMI core is not clocked by the HIGH phase of MCLK enveloped by nWAIT. Thus, during the cycles shown, nMREQ and SEQ change once, during the first LOW phase of MCLK, and A[31:0] change once, during the second HIGH phase of MCLK.

Absolute Maximum Ratings

Table 5. DC Maximum Ratings

Symbol	Parameter	Min	Max	Units
V_{DD}	Supply voltage	VSS – 0.3	VSS + 7.0	V
V_{in}	Input voltage applied to any pin	VSS – 0.3	VDD + 0.3	V
T_s	Storage temperature	-50	150	°C

Note: These are stress ratings only. Exceeding the absolute maximum ratings may permanently damage the device. Operating the device at absolute maximum ratings for extended periods may affect device reliability.

DC Operating Conditions

Table 6. DC Operation Conditions

Symbol	Parameter	Min	Typ	Max	Units	Notes
V_{DD}	Supply voltage	2.7	3.0	3.6	V	
V_{ihc}	IC input HIGH voltage	.8 x VDD		VDD	V	1,2
V_{ilc}	IC input LOW voltage	0.0		.2 x VDD	V	1,2
T_a	Ambient operating temperature	– 40		85	C	

1. Voltages measured with respect to VSS.
2. IC CMOS-level inputs.